Progress of C-Map: a Context-aware Mobile Assistant

Sidney Fels, Yasuyuki Sumi, Tameyuki Etani, Nicolas Simonet,
Kaoru Kobayashi, and Kenji Mase
ATR Media Integration & Communication Research Laboratories
Seika-cho, Soraku-gun, Kyoto, 619-02, JAPAN
fels@mic.atr.co.jp +81 774 95 1448 fax: +81 774 95 1408

Abstract

An interface agent with a life-like character on a personal mobile computer based guidance system is a plausible interface design approach for supporting and mediating communication between exhibitors and visitors of museums, laboratory tours and trade-shows. The guide agents should acquire their information ubiquitously and integrate this knowledge to provide context relevant assistant to visitors. C-MAP is a context-aware mobile assistant system that provides a mobile guide agent for assisting visitors during exhibitions. The system uses 30 portable computers, two servers, and an active badge location server. The portable computers are networked using a 1Mbit/sec wireless FM connection to the ethernet which the servers are connected to. Currently, the guide agent has a life-like animated character with four different behaviours, a physical map and a semantic map. The physical map tracks the visitor's location relative to the exhibits while the semantic map tracks the visitor's interests relative to the exhibitions. The guide agent uses the visitor's interest to plan a tour through the exhibition. A layered infrastructure is used to coordinate the guide agents and distribute their computation. This paper presents progress made towards the implementation of C-MAP.

1 Introduction

The concept of C-MAP, a context-aware mobile assistant was introduced by [8]. In working towards realizing the ideas of C-MAP considerable infrastructure has been created. This paper presents details of the network infrastructure and the progress made towards creating a guide agent.

As discussed in [8], when visiting a museum, exhibition or trade show, much of the information wanted by the visitor is a function of the context during the visit. For example, if they are at demonstration number 10, they would like to have information about demonstration number 10 available quickly. Further, if they express interest in what demo 10 is about, they will probably like to know if there are other related demos. One solution to this problem proposed in [8] is to have a tour guide who can interact with the visitor to understand what the visitor's interests are as the tour evolves. The C-MAP project is about trying to make a tour guide agent using portable computers, tracking hardware and computing servers.

An ideal personal tour guide does not need to be an expert in all the areas being exhibited; rather, he should be able to know how to connect a visitor with the relevant sources of information in a timely and contextually relevant manner. Further, he should also be able to gauge the interest of the visitor as she is shown exhibit discusses it with the tour guide and/or the exhibitor. The tour guide should be able to synthesize all this information along with any time constraints that the visitor has to make an effective visit to the exhibits. The tour guide should be able to assist the visitor in pursuing her interest after the exhibit is over. This may include introducing her to a community of people interested in the same things or providing directions for further investigation. Finally, guides should have their own sense of presence. That is, they should be members of the exhibition staff and be aware of the exhibition space and visitors in that space.

The main point about the ideal tour guide is that he must be able to adjust the information he provides in a contextually relevant way according to the evolution of the tour and the interests of the visitor. In our efforts to develop the ideas in [8], we used the annual open house exhibition of our research laboratory at ATR MI & C Research Laboratories as the first test bed for the C-MAP system. During the open house, the guide agent gathered information about the physical location of the visitor and the interests of the user to recommend demonstration sites to visit. Further, the guide agent provides services to assist the visitor in finding relevant demonstrations. These services include a physically based map so that visitors can see where they are relative to the demonstrations and a semantic based map so that visitors can see where their *interests* are relative to the demonstrations. The semantic based map is based on [9].

In implementing the ideas of C-MAP, we have two main focii; using an agent to integrate contextual information so that visitors have the information they need when they need it and fostering the development of communication links between people (either visitors and visitors or visitors and demonstrators). The first focus relates to some of the goals of ubiquitous computing [10]. The guide agent has the ability to detect the visitors locations in the tour. Further, the guide agent can generate only context relevant queries such as "Did you like that?" This ability allows the computing aspects of the guide agent to be hidden from the visitor since he will not have to choose from non-relevant information or attend to non-relevant queries.

There are other systems upon which the success of the C-MAP project is built. The ICMAS Mobile Assistant Project [6] provided a portable computer assistant that provided various services to assist conference attendees. The portability and wireless radio LAN access is similar to the C-MAP structure; however, connection was made using cellular phones. The attendees were responsible for connecting to the network to use email services and allow other services to use the network facilities. In contrast, C-MAP uses a 1 Mbit/sec wireless LAN to connect all the mobile computers together. This provides constant, unencumbering, high speed access to network services. This is important for a ubiquitous environment. The wireless aspect means that the machines can be with the visitor at all times. The constant access allows the mobile assistant to be constantly updated to any situation which may be relevant to its behaviour. The high speed access allows significant amounts of information, i.e., web pages to be loaded on demand with minimal wait by the visitor.

The Cyberguide [3] is also an attempt at providing context-aware mobile computing. Their hand-held tour guide was designed to assist visitors around their monthly open house at their lab as well. The main focus of the work revolved around using location-aware devices for context to assist visitors with a physical map. They used various TV remote control beacons which transmitted location identification to the hand held computers. The physical map and location context are the same as our system. However, we are attempting to capture visitor's interests. Currently, we do this by using the visitor's input when interacting with the semantic map. We also plan to monitor web page activity, engage in active dialogue and use on-site as well as off-site information about the visitor to assess their interests.

The Ubiquitous Talker [5] consists of a translucent LCD display that allows the visitor to see a scene through the display, a CCD camera for recognizing real world objects with color-bar ID codes, a microphone for recognizing human voice and a speech synthesizer. When the visitor looks at an object through the LCD, the CCD camera recognizes the bar code and provides relevant information about the object on the display and by speech synthesis. The system allows the visitor to feel they are speaking with the object itself. One of the goals of our system is to facilitate not only human-to-exhibit interaction, but also human-to-human interaction. That is, our view is that behind each exhibit are people who are knowledgeable and interested in it. The exhibit is a focal point for this community and the guide agent should help the visitor become part of it if appropriate.

2 Overview of C-MAP Implementation

The current implementation of C-MAP was built to provide a mobile computer based tour guide for visitors during the Open House exhibition at ATR M.I. & C. research laboratories. During the exhibition there were 19 different demonstration of research projects along with poster describing an additional 60 research projects going on at the research laboratories. At each demonstration there was at least one demonstrator to show the system. After registration at the C-MAP demonstration site, visitors were provided a portable computer which was their tour guide[1]. Figure 1 shows a visitor with one of the portable machines. Using one of the guide agents services, users could select from keywords to convey their interests. From this information, the tour guide agent can plan an itinerary and suggest demonstrations which may be of interest. The agent's display has three sections:

1. character animation representation of agent
2. physical map (location based relationship)
3. semantic map (semantic based relationship

The character animation had four different animations to indicate four agent states: thinking, suggesting, pushing and idle. The physical map showed the layout of the physical laboratory space along with marks for each demonstration and poster site. Additionally, the location of the visitor was marked. The physical map indicates the geographic distances between the visitor and the different sites. An active badge system from Olivetti was used to track user's locations. Finally, the semantic map was available which showed the semantic distances between the user's interest (indicated by keywords) and the various sites. Figure 3 shows an example of the display that the visitor sees. In this figure, only the animated character and physical map is visible. Due to the

[1] The tour guide processing was distributed between the portable machine and a server machine.

Figure 1: A visitor using one of the mobile assistants.

small screen size, the visitor was responsible for either scrolling or selecting between the the semantic map and the physical map.

A typical scenario for a visitor was the following. The visitor came to the registration desk. Registration personnel asked the visitor for some information to fill in a visitor profile. The visitor's profile included their name, company affiliation, initial interests, cartoon character they like, badge and machine name assigned. With this information an HTML page was generated. The assigned machine, running HotJava, loaded the newly created web page which started the tour guide agent. Once started, the tour guide agent contacted the central *agent server* and registered the badge and agent name with the server. The server uses this information to allow the active badge system to track the visitor's badge as they move through the laboratory. Additionally, the visitor's initial interest is communicated to the agent server so that an initial tour plan can be developed for this visitor. Recommended tour sites are highlighted on the physical map. As the visitor moves through the laboratory, their location is updated on the physical map. The visitor may also interact with the physical map and/or the semantic map using the portable machine's pointer (one portable used an isometric joystick and the other used a pen based input device). Moving the cursor over sites and clicking on sites caused information to be displayed about them.

When the visitor used the semantic map they could change the keywords that they thought were interesting. When this happened, the information was relayed back to the agent server so that a new plan could be determined. Further, the agent server is constantly aware of the location of the user so that it can use location history to plan new tour suggestions.

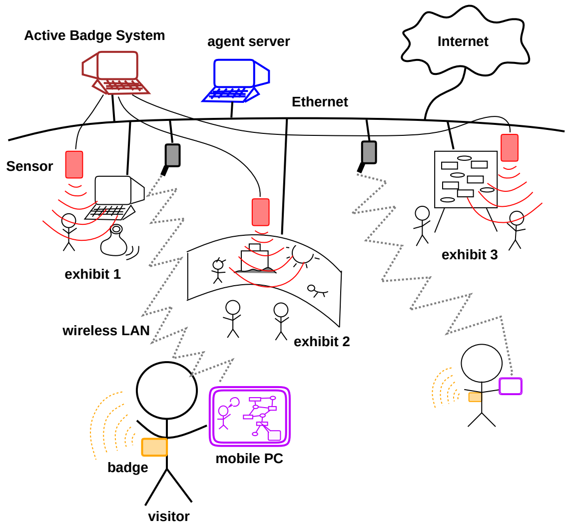

Figure 2: Schematic diagram of the C-MAP system. (Taken from [8])

Figure 3: Close-up of mobile agent's display. Only the physical map is shown here.

3 C-MAP Implementation

Figure 2 shows a schematic diagram of the C-MAP system. From the visitor's perspective, they are provided a portable computer with their personal guide agent. Figure 3 shows a typical display that a visitor would see on their portable guide agent machine. Notice, in the upper left corner is the animated character along with a message box which the animated character uses to send messages to the visitor. Also shown in figure 3 is the physical map. Notice that there are small markers showing the location of demonstrations and posters. On the same page as the physical map, shown in figure 4, is the semantic map. The semantic map shows semantic relatedness using Euclidean distance of keywords and objects.

The guide agent machines are connected to a central agent server using a 1 Mbit/sec wireless LAN. In addition, each visitor is wearing a badge that emits IR identification information. Sensors located at sites in the laboratory identify when a badge is in its sensor zone. The badge number and site location are communicated to the central server over an ethernet connection. In the following sections, the necessary hardware and software infrastructure required to connect the whole system is discussed. The implementation of the individual parts of the agent are briefly discussed.

3.1 C-MAP: Hardware and Software

The C-MAP system has three main pieces of hardware. The first is the portable computing machines to implement the display part of the agents. We use fifteen Libretto 60 computers and fifteen Amity VP computers. Each has 32 MB of memory and uses Windows95 and runs HotJava1.1 for web browsing. The second piece of hardware is the active badge system from Olivetti. This system tracks small badges worn by visitors using infrared signals detected by sensors placed strategically in the laboratory. The sensors are physically attached to a centralized PC. The badge identification and location are collected in a centralized machine and made available via the ethernet. The third piece of hardware is an agent server machine. Agents are connected to it by a wireless 1Mbit/sec FM LAN. For the agent server we use an SGI Onyx with 4 R4400 processors and 128 MB memory. Some of the agent's computing is performed on the agent server and some on the portables. The location server and agent server are physically attached to an ethernet and communicate using TCP/IP.

All of the agent code for the portable machines is in Java. We use HotJava1.1 as our browser since it allows the use of network services including sockets. The agent server is written entirely in Tcl [7] and its object oriented extension [incr Tcl] [4]. We use an in-house client/server protocol as described in section 3.3.

3.2 Guide Agent

Currently, the guide agent has four main components: the animated character, the physical map, the semantic map and the planner. The objectives for the guide agent during open house were fairly modest as we wanted to test the infrastructure before proceeding to more complex models of agent behaviour. The main objectives for the guide agent at this point were:

- track agentee's physical position on physical map
- track agentee's interests on semantic map (using keyword input to measure interest)
- suggest alternative demonstrations sites to visit depending upon agentee's interest (which may change throughout the visit)
- provide an animated character which represents the guide agent's internal state

For each agent, the planner is implemented on the agent server machine using [incr Tcl] [4]. The other components are implemented on the portable machine in Java. The parts of the agent implemented on the portable computers are called the p-agent components and the parts implemented on the fixed machine are called the f-agent components. A communication infrastructure has been constructed to allow the p-agent to communicate with its f-agent as well as the agent server as described in subsection 3.3. The same infrastructure is used to communicate with the badge location server.

We partially used an Asynchronous Hierarchical Architecture (A-HA) [1] to implement our guide agents. In this framework, the agent's knowledge and control structures are hierarchical consisting of a motivation, policy, behaviour and action layer. Further, all activities which may occur both inside the agent and in the agent's world are asynchronous. By using the A-HA approach, the guide agent responds to events as they happen. Due to time constraints, our current implementation of the planner is very simple and operates at the behaviour layer. Our future work deals with increasing the complexity of our agents to use the full hierarchy and distributing the computational load.

The following sections describe each of the components implemented on the portable machine (p-agent components). These components along with the badge location server information are the guide agent's main source of information about the agentee. Further, these are the mechanisms which the agent uses to transmit information to the agentee.

3.2.1 Animated Character

The animated character is the simplest component of the p-agent components. We created 11 different characters which the visitor could choose from. Each character had short animations of four different behaviours which could be controlled remotely (see section 3.3). The four different behaviours included thinking, suggesting, pushing and idle. Additionally, the animated character could be set to exhibit a particular behaviour for given duration. The behaviours were implemented as a FIFO queue. When the queue is empty the idle behaviour is displayed. The animated character applet is responsible for sending the agent server the name of the animated character the visitor chose when they registered. Above the animated character was a small message window which can also be controlled remotely.

3.2.2 Physical Map

The physical map is shown in figure 3. The map shows the physical layout of the laboratory. Additionally, location and attributes of demonstration and poster sites can be dynamically added and changed remotely (see section 3.3). Further, the location of the agentee can be indicated dynamically. The physical map allows the agentee to zoom in on interesting areas. Each location has attributes such as colour, description of site, URL, etc. The site description appears when the agentee moves the mouse over a site and pauses. When the user clicks on the site, the URL is loaded into a new web browser[2].

The physical map is controlled by the agent server whenever the visitor moves to a new demonstration site. As soon as one of the sensors detects a badge, the badge id and the sensor location is transmitted to the agent server, which then marks the visitors location on the map. Further, the planner controls the look of the physical map to suggest different sites that the visitor should go to. The physical map is responsible for sending the agent server the badge number of the current visitor after registration.

3.2.3 Semantic Map

The semantic map is shown in figure 4. The map shows the semantic distance between objects (based on keywords) transformed to a Euclidean space. The objects can be people, demonstration sites or keywords. When the visitor is using the semantic map they can choose keywords which indicate their interests. These keywords are located on the semantic map along with icons representing people, demonstrations and posters which are in the lab. The visitor can then see which objects are most closely related to the keywords they have chosen.

The selected keywords are used to form an interest vector. The interest vector is a list of all demonstration sites ordered by their relatedness to the visitor's selected interest keywords. This interest vector is sent to the agent server where the agent's planner is implemented. The planner then uses this information along with location information and history to suggest which demonstrations should be seen next. Further, the agent server also uses location information to highlight demonstration objects in the semantic map when visitors enter the demonstration site's sensor zone.

[2]This feature was not used during open house. The loading of a URL can also be controlled remotely.

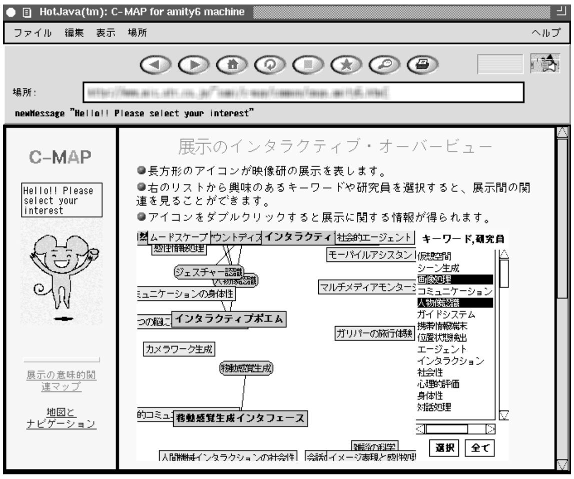

Figure 4: The guide agent's display showing the semantic map.

3.3 Client/Server Communication Infrastructure

A key component for C-MAP to work is the communication infrastructure. The general structure is shown in figure 5. The agent server and the clients have a layered structure for communicating with each other. There are two main ways this structure is used

1. agent clients use agent server's services (including executing the f-agent part of the agent)
2. agent server controls agent client

In our framework, the Tcl layer of the agent server accepts strings sent to it and executes them as Tcl scripts. On top of this layer is an object created using [incr Tcl] which is the agent server layer. This object provides public methods which the agent clients can run remotely. Thus, when an agent client wants to use an agent server's services it sends a Tcl script to execute one of the agent server's methods. In the same manner, when a component in the p-agent wants to execute methods on the f-agent it sends a Tcl script to the agent server which runs the appropriate method.

As an example, consider when a user finishes registering. At this point, a new HTML page is created for them which is loaded into HotJava on the portable machine. This page has the registration information for the new visitor. When HotJava loads the page, the agent is started but still needs to send the registration information to the server. To see why the agent server needs this information, consider the visitor's badge number. The sensor site location coordinates of each badge are determined by the active badge system and transmitted to the agent server. The server needs to know which agent is currently using this badge so that it can update the position of the visitor on the agent's physical map.

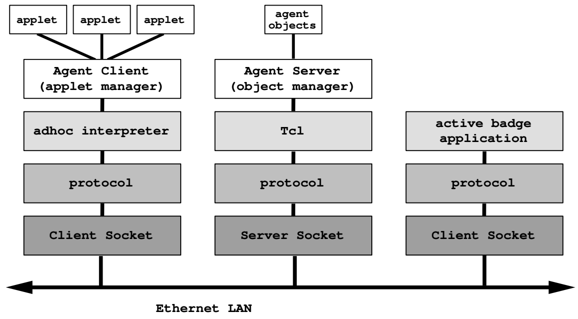

Figure 5: Schematic diagram of the C-MAP infrastructure

Thus, the agent must send its own name and badge number to the agent server. To do this the agent sends the following Tcl string:

```
%S registerBadge agent1 Badge10
```

The %S says run the registerBadge command on the agent server with the two arguments: agent1 and Badge10[3]. The other Tcl commands the agent sends are:

```
%S registerCharacter agent charName
%S setInterest "information vector"
```

which register the type of animated character to use and the list of prioritized demonstration sites determined by the semantic map for the planner, respectively.

The active badge system uses a similar technique to communicate badge locations as they are determined. When a new badge location arrives, the active badge layer in figure 5 sends a Tcl string to the agent server with the badge number the name of sensor which detected the badge. The agent server has a data base with sensor names and the X-Y location of the sensor in the laboratory.

3.3.1 Applet Interpreter Interface

Currently, the agents on the portable systems only have a application specific interpreter. This adhoc interpreter layer implements an interpreter which allows application specific commands to execute Java code. We call the adhoc interpreter a Java Interpreter. When the protocol layer of an agent receives a message with a Java interpreter header it passes the string along to the Java interpreter. The Java interpreter then parses the message and calls the appropriate method maintained by the agent client layer in the specified applet. The applets correspond to one of the three functions that the agent has: animation applet, physical map applet or semantic map applet. This communication mechanism is easily illustrated by example. Consider the animation applet. It has four animations to exhibit four different behaviours. The agent client has a command:

```
addBehaviour THINKING 10
```

If the agent running on the agent server will be thinking for 10 seconds it can send the above command to the agent. The agent server's protocol layer knows that the destination agent has a Java Interpreter and assembles the message accordingly and sends it to the portable agent. When the portable agent receives the message, its protocol layer recognizes that a Java Interpreter command has been received and passes it along to the interpreter which parses the message and calls the appropriate method in the agent layer along with the two arguments. Each applet has different publically accessible methods which are available interactively by using the Java Interpreter. In this way, different markers on the physical map or semantic map can be added or have their attributes changed. To illustrate, here are some the commands available in the Java Interpreter:

1. Animation applet
   - **addBehaviour** behaviour time
   - **resetBehaviour** [behaviour] [time]
   - **newMessage** message
2. Physical Map applet
   - **addDemo** name x y
   - **addPoster** name x y
   - **setPosition** componentName x y
   - **setState** componentName stateName (Highlight, Normal, Blurred)
   - **highlightDemo** demoNumber
   - **setAllState** stateName
   - **setURL** componentName URL
   - **setPopupInfo** componentName info
   - **switchTo** applet
3. Semantic Map applet
   - **highlightDemo** demoNumber

To summarize, agent clients send Tcl scripts to the agent server and the agent server sends Java interpreter commands to the agent. The agent server is an [incr Tcl] object. Since it is an [incr Tcl] object, agents can manipulate it by executing Tcl scripts in the Tcl interpreter

[3] The %S actually is a special flag which tells the protocol layer to replace the flag with the name of the agent server that this agent is connected to.

where the agent server is. The Tcl layer provides a communication channel from the network to the Tcl interpreter where the agent server is; thus, agents can send Tcl scripts to the agent server's machine and the Tcl layer will execute them. The agents can execute any Tcl script in this manner; however, the when an agent connects to an agent server the agent server's name is passed to the agent layer to use. The agent server layer then restricts applets to only run agent server methods. We plan to replace the adhoc interpreter in the client to a Tcl interpreter implemented in Java [2]. This will allow the agent server to control the agent client using Tcl scripts too.

One advantage of the communication infrastructure used in C-MAP is that agents can easily be distributed onto any machine required. Further, the computation can be though of in terms of objects relaying messages between themselves. The messages sent are just Tcl scripts. For example, in the C-MAP application we plan to implement objects which represent demonstrations, demonstrators, visitors, agents (using A-HA architecture) and a world activity manager. These objects can be implemented in C, C++, Java or Tcl. Each object will be able to seamlessly interact with any other object using the communication structure currently developed.

This architecture has a significant advantage when interfaces to other applications and exhibit objects are needed. For example, a demonstration object may want to retrieve context information about the current visitor. These demonstrations can then adjust themselves to suit the visitor. In the open house we had two such demonstrations. First, was a 3D walkthrough demonstration of the laboratory. All the badge's locations were continually sent to this application so that all the visitors could be placed inside the 3D scene. Second, we had a 3D walkthrough of a simulation of a archeological dig. When the visitor used the system, their animated guide character appeared in the walk through.

4 What Worked and What Didn't

The C-MAP project has many parts which all have to work for the system to function. Further, these parts must be well integrated, provide consistent service and seamless operation for the visitor. By the time open house started the following parts were working properly:

- Each of the applets worked well independently and provided an interface to the agent layer for external control. The interface implemented for each applet was minimal as can be seen from section 3.3.1.
- The client/server infrastructure was completed and functioned well. During the 2 days of open house, when all 30 machines were running we experienced only one unrecoverable error.
- The planner part of the agent was simple but worked. The planner was able to highlight recommended demonstrations for the visitor using the information vector coming from the semantic map. However, the planner did not use any location history information.
- The active badge output was integrated into the C-MAP system. This allowed the agent server to attend to changes in badge location when they occurred.
- The registration procedure went smoothly. Gathering visitor's profiles and assigning badges, starting up machines and starting the guide agent worked well.

Some of the parts of C-MAP did not meet our expectations. In particular:

- The active badge system from Olivetti advertised that 32 sensors could be used and spread over the demonstration sites. In fact, it was not capable of monitoring more than 6 sensors. Because of this, we had to limit our experiment to six demonstration sites. Further, the update rates sometimes had lags of up to 5 seconds. This made the system be up to 5 seconds out of date. In a small space, such as our lab, this can be problematic for providing timely information.
- The animation character was able to display different states of the agent. However, since the planner was not too complex there were no states to display.
- The planner did not use location information and visitor interaction to form its plan.
- The guide agent did not allow web surfing during the tour. It was planned that for each demonstration (and poster) site there would be a web page which would appear when the visitor was in front of it. This was prepared but not made available for the visitors.
- Only modest integration of each of the three parts was done. The external interface to each of the three parts was not sufficient to support a high degree of integration. Further, visitor's interactions with each of the different applets was not used to determine visitor's interest in various demonstrations and posters.

5 Future Work

At this point, the C-MAP system experimented with at ATR's open house is still in the early stages of development. Now that much of the infrastructure has been built which allows agents running on the portable machines to communicate nearly seamlessly with an agent server (and each other) we are improving the agent side of C-MAP.

The main areas we are working on for the C-MAP system include:

1. Increasing the complexity of the agent to include more context from dialogue with the visitor and visitor interaction statistics.
2. Providing other guide agent services including match making assistance and community development facilitation.
3. Constructing a demonstration agent to help visitors interact with demonstrators.
4. Get a better tracking system. It would be ideal to have a tracking system that can give not only site locations, but also X,Y coordinates of the visitor on the floor. Further, the sample rate should be as fast as possible with as little lag as possible.
5. Improving the communication infrastructure by replacing the adhoc interpreter with a Tcl interpreter.

6 Conclusions

The first test of C-MAP was successful in the sense that we were able to build and investigate the C-MAP infrastructure and see where technical problems were. The infrastructure worked well and was reasonably flexible allowing last minute changes to be integrated easily. The guide agent is still relatively immature and only has simple context awareness. We are currently improving the guide agent and integrating more context for the guide agent to make inferences about the interests and needs of the visitor.

Ultimately, we hope to have a guide agent that can facilitate communication between real-world objects (exhibits), exhibitors, virtual knowledge spaces (such as web sites) and visitors. For this to work it is important that we can track and assess the user by unobstructively monitoring them and using this information. From the visitor's perspective, this information gathering will be invisible, but the resulting guide agent behaviour will be adapted to the task at hand. By using appropriate contexts and integration of various media the line between virtual reality and real reality will become invisible.

7 Acknowedgements

Our project leader, Yasuyuki Sumi, was instrumental in procuring the gratious cooperation of fellow research members in the M.I.&C. Laboratories while we disturbed them to gather information needed for this project. Their patience is greatly appreciated.

References

[1] A. Bruderlin, S. Fels, S. Esser, and K. Mase. Hierarchical agent interface for animation. In *Animated Interface Agents Workshop Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'97)*, August 1997.

[2] I. Lam and B. C. Smith. Jacl: A Tcl implementation in Java. In *Proc. 5st Tcl/Tk Workshop*, pages 31–36, Boston, CA, USA, 1997.

[3] Sue Long, Dietmar Aust, Gregory Abowd, and Chris Atkeson. Cyberguide: Prototyping context-aware mobile applications. In *CHI 96 Conference Companion*, pages 293–294. ACM, 1996.

[4] M. McLennan. [incr Tcl]: Object-oriented programming in Tcl. In *Proc. 1st Tcl/Tk Workshop*, University of Berkeley, CA, USA, 1993.

[5] Katashi Nagao and Jun Rekimoto. Ubiquitous talker: Spoken language interaction with real world objects. In *IJCAI-95*, pages 1284–1290, 1995.

[6] Yoshiyasu Nishibe, Hiroaki Waki, Ichiro Morihara, and Fumio Hattori. Analyzing social interactions in massive mobile computing –experiments of icmas'96 mobile assistant project–. In *IJCAI-97 Workshop on Social Interaction and Communityware*, pages 19–24, 1997.

[7] J. K. Ousterhout. *Tcl and the Tk Toolkit.* Addison-Wesley, New York, 1994.

[8] Yasuyuki Sumi, Tameyuki Etani, and Kenji Mase. Context-aware mobile assistant. In *Proceedings of the 55th IPSJ Annual Meeting*, pages 443–444, Sept. 1997.

[9] Yasuyuki Sumi, Koichi Hori, and Setsuo Ohsuga. Computer-aided thinking by mapping text-objects into metric spaces. *Artificial Intelligence*, 91(1):71–84, 1997.

[10] M. Weiser. Some computer science issues in ubiquitous computing. *Communication of the ACM*, 36(7):75–83, July 1993.